

## АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА ПРАЗНИЧНО ПЪТУВАНЕ

Задачата решаваме с модификация на алгоритъма на Floyd. Освен матрицата  $G$  в която въвеждаме графа във вида матрица на съседства, и в която алгоритъмът на Floyd намира дължините на най-късите пътища от всеки връх до всеки друг връх, използваме още една матрица  $P$ , в която за всеки два върха ще намерим броя на най-късите пътища от всеки връх до всеки друг връх. Преди началото на алгоритъма, нулираме всички елементи на матрицата  $P$ .

Когато алгоритъмът на Floyd установи истинността на проверката в релаксационната стъпка:

```
if (G[i][k]+G[k][j]<G[i][j])
```

това означава, че са намерени един или повече по-кратки пътища между върховете  $i$  и  $j$  (от този, чиято дължина е запомнена до момента), минаващи през връха  $k$ . Затова, освен че заменяме старата запомнена дължина с новата:

```
G[i][j]=G[i][k]+G[k][j];
```

трябва да запомним в  $P[i][j]$  броя на тези по-къси пътища. Тъй като в  $P[i][k]$  е запомнен броят на временно най-късите пътища от  $i$  до  $k$  и същото е в сила за  $P[k][j]$ , то броят на най-късите пътища от  $i$  до  $j$  ще бъде произведението на  $P[i][k]$  и  $P[k][j]$ :

```
P[i][j]=P[i][k]*P[k][j];
```

За разлика от оригиналния алгоритъм на Floyd, обаче, тук трябва да проверим още една възможност:

```
if (G[i][k]+G[k][j]==G[i][j])
```

и ако това условие се окаже изпълнено, към намерените вече най-къси пътища, минаващи само през върхове с номера по-малки от  $k$ , чиято дължина е запомнена в  $G[i][j]$ , трябва да се добавят и новите пътища със същата дължина, минаващи през  $k$ :

```
P[i][j]+=P[i][k]*P[k][j];
```

Ето и цялото решение на задачата:

```
#include <iostream>
#define MAXN 600
#define MOD 1000000021
#define INF 1000000000
using namespace std;

int G[MAXN][MAXN], N, M, Q;
long long P[MAXN][MAXN];

int main()
{
    ios::sync_with_stdio(false);
    cin.tie(NULL);
    int i, j, k, u, v, c;
    cin >> N >> M;
    for (i=1; i<=N; i++)
        { for (j=1; j<=N; j++) {G[i][j]=INF; P[i][j]=0;}}
```

```

    G[i][i]=0;
}
for(i=1;i<=M;i++)
{  cin >> u >> v >> c;
    G[u][v]=G[v][u]=c;
    P[u][v]=P[v][u]=1;
}
for(k=1;k<=N;k++)
    for(i=1;i<=N;i++)
        for(j=1;j<=N;j++)
            if(G[i][k]+G[k][j]<G[i][j])
            {  G[i][j]=G[i][k]+G[k][j];
                P[i][j]=(P[i][k]*P[k][j])%MOD;
            }
            else if(G[i][k]+G[k][j]==G[i][j])
                P[i][j]=(P[i][j]+P[i][k]*P[k][j])%MOD;
cin >> Q;
for(i=1;i<=Q;i++)
{  cin >> u >> v;
    if(u==v) cout << 1;
    else cout << P[u][v];
    if(i<Q) cout << " "; else cout << "\n";
}
return 0;
}

```

*Автор: Красимир Манев*